ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-TUPHA171

# DEVELOPMENT OF NICA CONTROL SYSTEM: ACCESS CONTROL AND LOGGING

Evgeny V. Gorbachev, Georgy Sergeevich Sedykh, JINR, Dubna, Russia

*Abstract*

NICA (Nuclotron-based Ion Collider fAcility) is a new accelerator complex being constructed at the Joint Institute for Nuclear Research (Dubna, Russia). It will provide heavy ion colliding experiments to study properties of dense baryonic matter. The TANGO based control system of the NICA complex is under development now. The report describes design of the role-based authorization and logging system. It allows limiting access to any Tango device command or attribute according to a user roles and location. The system also restricts access to the Tango database and records details of its modifications. The authorization is performed on the Tango server side thus complementing the native TANGO client-side access control. First tests of the system were performed during the latest Nuclotron run.

## INTRODUCTION

NICA accelerating complex is currently under construction in Joint Institute for Nuclear Research, Dubna, Russia. The complex will consists of heavy-ion and polarized particles sources, RFQ injector, heavy- and light-ion linear accelerators, superconducting booster synchrotron, Nuclotron and two superconducting collider rings [1]. The first stage of the accelerator complex will start its operation in the end of 2018 year in start-up configuration which includes full injection complex, Booster synchrotron, upgraded Nuclotron synchrotron and BM@N detector. The final configuration of the NICA complex including NICA collider and MPD detector will start the operation in 2023 providing design luminosity of $1.0\cdot10^{27}$ cm$^{-2}$s$^{-1}$ in $^{197}$Au$^{79+}$ collisions.

The TANGO controls [2] based control system is being developed at JINR [3]. The control system of the NICA complex aims at accomplishing few main tasks:

- Management of large amount of equipment which is distributed on the accelerator complex area.
- Realization of different regimes of the accelerator complex working cycle – colliding or fixed target experiments, various ion types and energy.
- Strict synchronization of accelerators in the chain.
- Comprehensive beam diagnostics during the entire cycle.
- Providing protection and safety measures.

## ACCESS CONTROL SYSTEM

One of the most important aspects of the control system operation is restriction of access to the control system components. It is necessary to allow access to control system software and hardware only for certain users and give them corresponding rights to perform specific tasks. Other users have to be provided with status information without ability to control the equipment.

The access restrictions can be implemented either on network level by using private networks and certain firewall configuration or on the software level. The most effective method is to combine network and software restrictions.

The software checks in TANGO control system can be implemented by:

1. Checking access control in client applications. The operator user interface can enforce an additional authentication for performing some critical operations and perform logging of the operator actions. However, this type of access control cannot prevent usage of the other client applications such as standard TANGO client ‘jive’.
2. Using native TANGO access control. It works on the client side and can restrict access to TANGO devices to certain system accounts on certain IP addresses. However, usage of system accounts limits the flexibility of the system. Also, the native TANGO access control cannot be used for the authorization of Web clients because all the requests originate from one or few REST TANGO devices running on the fixed IP addresses [4].

An additional access control system is being developed at JINR to overcome limitations of the existing solutions for a TANGO based control system. The idea is to give a TANGO device ability to check the client’s permission to execute device’s commands or access attributes. We would like to have a central database of users, their permissions and global access logs. It’s necessary to provide a simple way to implement authorization on the TANGO device side without modification of TANGO libraries. Also, the system has to implement a way to check permissions of the Web applications which communicate with TANGO control system via REST TANGO devices.

### *Authorization Algorithm*

The proposed operation algorithm of the authorization system is shown in Fig.1. Here, a TANGO client tries to execute a command on a TANGO device. The operation of the access control system is provided by two additional TANGO devices: the authorization TANGO device and the authorization manager TANGO device.

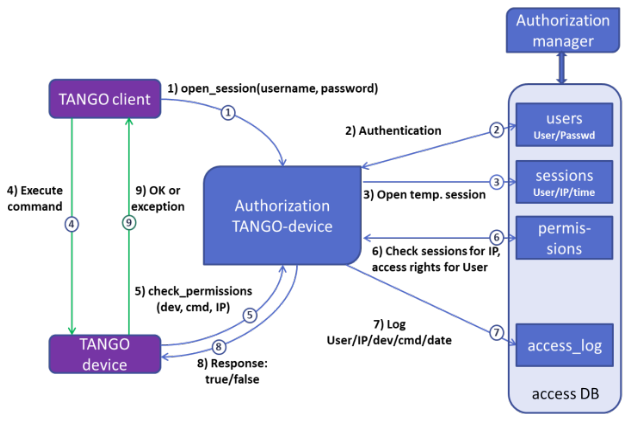


Figure 1: TANGO server-side authorization algorithm.

The process of the authentication and authorization is split into few stages:

1. The TANGO client executes the *open_session()* command on the authorization TANGO device providing username and password. In the current implementation, this can be done from separate process running on the same IP address.
2. The authorization device authenticates the user against the access control database.
3. In the case of successful authentication, the authorization server opens a temporary session saving the username, the client's IP address and the current time in the 'sessions' table.
4. The TANGO client executes the command on the TANGO server using standard TANGO client API.
5. The TANGO servers executes command *check_permissions()* on the authorization TANGO device providing device name, command name and client's IP address.
6. The authorization TANGO device checks the 'sessions' table for the sessions opened from the given IP address, gets the authenticated username(s) and checks their permissions against the 'permissions' table.
7. The information about username, IP address, device name and command name is written to the 'access_log' table.
8. The authorization result is returned to the TANGO device.
9. The TANGO device either executes the command and returns its results to the TANGO client or generates an exception.

It should be noted, that TANGO client is not providing any information except for the username and password. The client's IP address is obtained by authorization server from CORBA connection data by using *Tango::DeviceImpl::get_client_ident()* method.

The authorization manager TANGO device is intended for the administration of the access control database.

## *Realization Details*

The authorization system implements Role-based access control (RBAC) approach: every user is assigned a number of roles that he can use from certain IP addresses. Each role consists of a number of permissions to access TANGO devices commands and attributes.

The RBAC system uses MySQL database with InnoDB storage engine. The structure of the database is shown in Fig. 2. The database has few tables which are linked together by means of foreign keys. The foreign keys constraints help keeping the spread-out database data consistent: for example, the access cache entries are automatically deleted when the user session expires and the session entry is cleared out of the session table. Also, MySQL triggers are used to clear the access cache when the permission table changed. It helps to enforce the update of user's permissions immediately after the permissions table change.

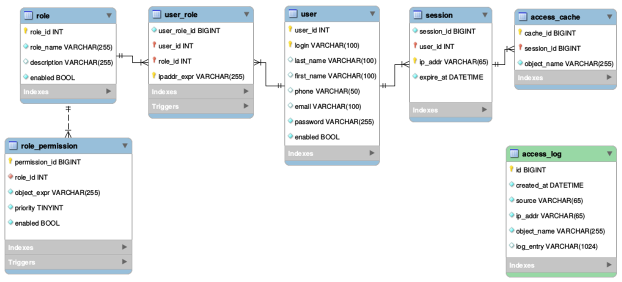


Figure 2: Access control database structure.

Permissions have priorities that allow enforcing usage of expert account to perform some critical operations. The priority of level 0 has the highest precedence.

It is possible to use either MySQL regular expressions or wildcard expressions in the permissions or IP address specifications. It can be configured in the authorization TANGO device property. Typical permission expressions with MySQL wildcards may look like: '*sys/database/%/DbGet%*', '*satellites/opc/3/%*' and so on.

There is a default user 'operator' which has no password. Some roles can be assigned to this user to be executed from certain IP addresses. It allows organizing authorization by location which is useful for providing control room computers with necessary permissions.

The access cache table is used to store the successfully authorized access requests. It significantly improves the authorization server performance especially when using regular expressions.

## *Authorization Tango Device*

The authorization TANGO device provides a number of commands to implement authentication and authorization procedures:

1. Authentication commands used by TANGO clients:
   a) *open_session(username, password)* – the authorization TANGO device checks the provided username and password against the database and opens the temporary session.
   b) *close_session(username, password)* – the client can provide username and password to force the session close.

2. Authorization commands used by TANGO devices to check the client permissions:
   a) *check_permissions(device_name, cmd_name, IP)* – the authorization TANGO device checks if the client from the given IP address is authorized to execute the command on the TANGO device. The TANGO device obtains the client's IP address from the CORBA connection data by using *Tango::DeviceImpl::get_client_ident()* method.
   b) *check_permissions_www(username, password, device_name, cmd_name, IP)* – the command is used by TANGO REST devices to authenticate and authorize Web clients. The REST TANGO device obtains the client's IP address either from socket connection or from HTTP header 'X-Forwarded-For' added by Apache proxy module and containing the IP address of the client.

The performance of the TANGO RBAC authorization server was tested by measuring the latency of sequential executions of the *check_permissions()* command. The results are shown in Fig. 3. It can be seen, that regular expressions are much slower than wildcards. However, the access cache speeds up all subsequent executions of permissions checks.

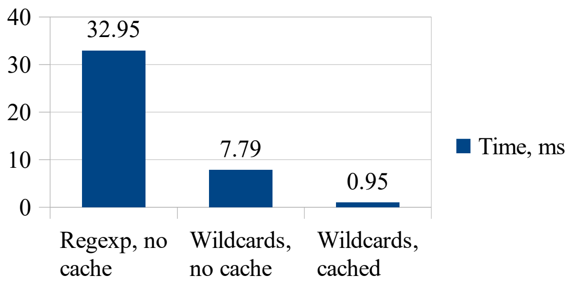


Figure 3: The authorization TANGO device performance.

## *Authorization System Usage*

The high level C++ class was developed to simplify the usage of the access control system. The TANGO device developer can implement the authorization by few simple steps:

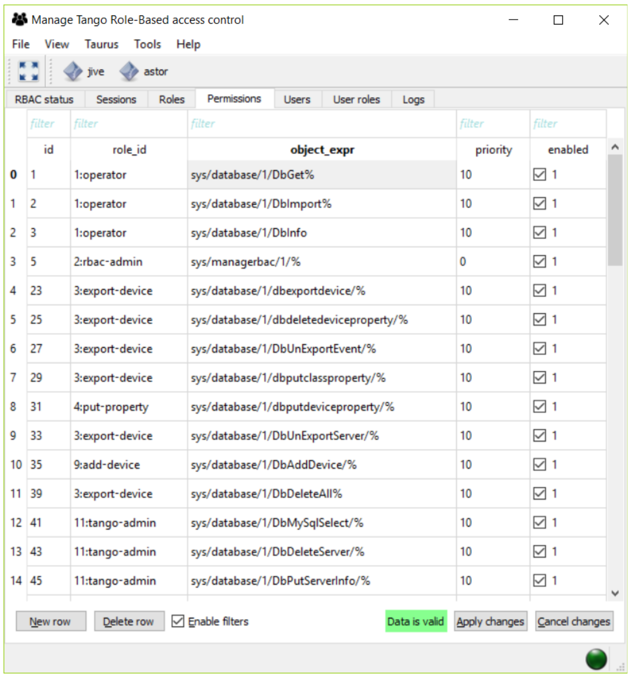
1. Include the header file:
   *#include "TangoAuth.h"*
2. Instantiate the class in *init_device()* method:
   *auth=new TangoAuthClientClass(this);*
3. Perform the authorization check in the TANGO command implementation code:
   *auth->CheckAccess("cmd_name");*

Also, the authorization check can be executed from *always_executed_hook()* method to protect all the available commands and attributes.

## *Management of the Access Control System*

A special TANGO device was developed to administer the access control database. It has a number of commands to manage all aspects of the system: adding or removing users, roles, permissions and retrieving the logs.

The graphical client was created using python and Taurus to give administrators convenient interface to manage the system. Its main window with permissions is shown in Fig. 4.


Manage Tango Role-Based access control
File View Taurus Tools Help
jive astor
RBAC status Sessions Roles Permissions Users User roles Logs
New row
Delete row
Enable filters
Data is valid
Apply changes
Cancel changes

| | id | role_id | object_expr | priority | enabled |
|---|---|---|---|---|---|
| | filter | filter | filter | filter | filter |
| 0 | 1 | 1:operator | sys/database/1/DbGet% | 10 | ✓ 1 |
| 1 | 2 | 1:operator | sys/database/1/DbImport% | 10 | ✓ 1 |
| 2 | 3 | 1:operator | sys/database/1/DbInfo | 10 | ✓ 1 |
| 3 | 5 | 2:rbac-admin | sys/managerbac/1/% | 0 | ✓ 1 |
| 4 | 23 | 3:export-device | sys/database/1/dbexportdevice/% | 10 | ✓ 1 |
| 5 | 25 | 3:export-device | sys/database/1/dbdeletedeviceproperty/% | 10 | ✓ 1 |
| 6 | 27 | 3:export-device | sys/database/1/DbUnExportEvent/% | 10 | ✓ 1 |
| 7 | 29 | 3:export-device | sys/database/1/dbputclassproperty/% | 10 | ✓ 1 |
| 8 | 31 | 4:put-property | sys/database/1/dbputdeviceproperty/% | 10 | ✓ 1 |
| 9 | 33 | 3:export-device | sys/database/1/DbUnExportServer/% | 10 | ✓ 1 |
| 10 | 35 | 9:add-device | sys/database/1/DbAddDevice/% | 10 | ✓ 1 |
| 11 | 39 | 3:export-device | sys/database/1/DbDeleteAll% | 10 | ✓ 1 |
| 12 | 41 | 11:tango-admin | sys/database/1/DbMySqlSelect/% | 10 | ✓ 1 |
| 13 | 43 | 11:tango-admin | sys/database/1/DbDeleteServer/% | 10 | ✓ 1 |
| 14 | 45 | 11:tango-admin | sys/database/1/DbPutServerInfo/% | 10 | ✓ 1 |

Figure 4: Access control system administration client.

## *TANGO Database Protection*

TANGO database is the most important component of the TANGO-based control system. It is desirable to restrict access to the database and store the log of the database changes. The TANGO database server implements a TANGO device which provides access to the TANGO MySQL database through a set of commands.

The TANGO database protection can be done either by rewriting of the database TANGO device server with addition of the authorization checks before each command method or by creating the TANGO device in the middle that implements the authorization, executes the command on the original TANGO database device and then return the result to the client or generate an exception.

The latter approach was accomplished by creating an alternative TANGO database device (see Fig. 5).

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-TUPHA171

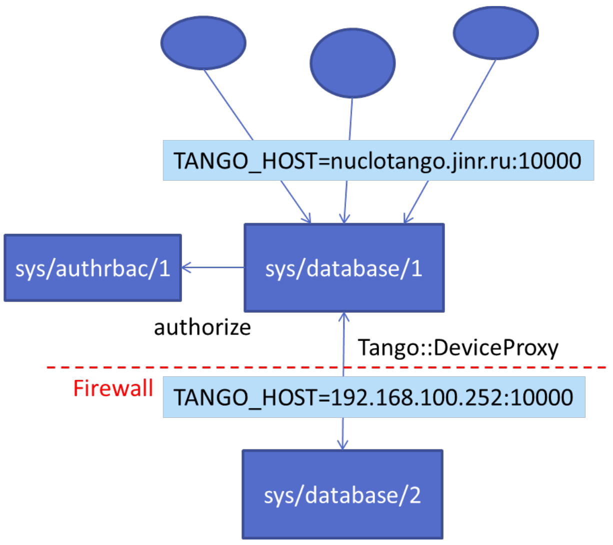


Figure 5: The TANGO database protection.

The following algorithm was realized:

1. Initialize as TANGO database server with NO_SYNC serialization for the best performance:

*Tango::Util::_UseDb = false;*
*set_serial_model(Tango::NO_SYNC);*

2. Get the list of the original TANGO database device commands and attributes:

*dbdev = new Tango::DeviceProxy("sys/database/2");*
*cmdlist=dbdev->get_command_list();*
*cmdinfo=dbdev->get_command_config(cmdlist);*

3. Create dynamic commands using the obtained configuration. All the commands inherits the same command class with method *execute()* which performs:
   - client authorization via TANGO authorization device;
   - execution of the command on the original database device;
   - return the command execution results to the client or generate an exception.

The original TANGO database device is protected by firewall and not accessible by TANGO clients.

The implemented access control allows restricting execution of the TANGO database device commands, for example, one can specify computers which can export TANGO devices, add or modify TANGO devices properties and so on.

The additional authorization and logging makes the communication with TANGO database slower. The TANGO database device performance was measured using Apache JMeter [5]. The test was performed by executing a command on the original TANGO database device and the TANGO database device with access control. The average duration of the command execution with different threads number on both TANGO devices is shown in Fig. 6.

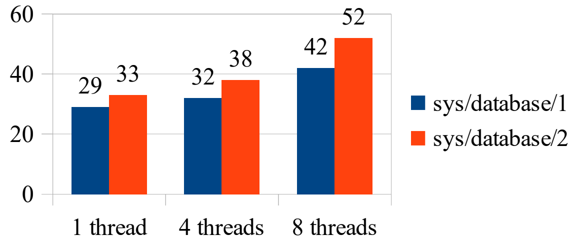


Figure 6: Database devices performance tests.

It can be seen that TANGO database server with authorization support is slower than original one but the difference is not huge – about 25% in the worst case.

## LOGGING

All authentication and authorization requests and their execution status are written into the access_log table of the access control database. This information can be very useful allowing reconstructing the flow of TANGO commands and their origin in case of problems in the control system.

Besides that, the authorization TANGO device provides the command *log_message(device, cmd, IP, message)*, which allows a TANGO device to write important information into the log table. The TANGO device has to be authorized to be able to write the messages.

## CONCLUSION

A role based access control system for the NICA control system was designed. It allows a TANGO device to check client's permissions to execute a command or access an attribute. The authorization system uses additional TANGO device which provides necessary command interface to RBAC database to manage authentication and authorization.

The system has been running on the Nuclotron control system since beginning of 2017 and proved to be useful. It was successfully used for limiting access to TANGO devices and database. The information in the access log helped to find and fix some problems in the control system – several TANGO devices produced extensive writes into the TANGO database.

The source code of the software is available at http://tangodevel.jinr.ru/git/tango/auth.

## REFERENCES

[1] G.Trubnikov, N.Agapov, O.Brovko at al., NICA project at JINR, Proceedings of IPAC2013. – Shanghai, China, 2013

[2] TANGO Controls, http://www.tango-controls.org

[3] V. Andreev, E. Gorbachev, A. Kirichenko et al., Nuclotron and NICA control system development status, Proceedings of ICALEPCS2015. – Melbourne, Australia, 2015, Pp. 437-440

[4] G.Sedykh, V.Elkin, E.Gorbachev, Tango Web Access Modules and Web Clients for NICA Control System, ICALEPCS2017, TUPHA167.

[5] Apache Jmeter, http://jmeter.apache.org